Claims:

1. A data management system in a computing environment comprising:

a. a data instance centric architecture;

b. where each data instance is encapsulated in a common fundamental data structure; and

c. where said common fundamental data structure also encapsulates references to associated separately encapsulated data instances.

2. The data management system of claim 1 wherein the said data-instance centric architecture and the said common fundamental data structure have structural symmetry.

3. The data management system of claim 1 wherein a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance.

4. The data management system of claim 3 wherein said data-instance centric architecture and the said fundamental data structure and the said encapsulated data instances and references have structural and relationship symmetry.

5. The data management system of claim 1 wherein a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance;

wherein each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances;

and

wherein said logical index is 'm' dimensional, and has 'n' bits per dimension.

6. The data management system of claim 5 wherein said data instance centric architecture and said fundamental data structure; and the said encapsulated data instances and references have structural, relationship, value and containment symmetry.

7. The data management system of claim 1 wherein:

said encapsulated references are in at least one dimensions; and

each of said at least one dimensions corresponds to a type of association.

8. The data management system of claim 7 wherein each of said at least one dimensions has a plurality of said encapsulated references.

9. The data management system of claim 1 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

10. The data management system of claim 2 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

11. The data management system of claim 3 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

12. The data management system of claim 4 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

13. The data management system of claim 5 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

14. The data management system of claim 6 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

15. The data management system of claim 7 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

16. The data management system of claim 8 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

17. The data management system of claim 1 wherein at least one of said encapsulated references is a reference to an encapsulated data instance in another computing environment.

18. The data management system of claim 1 wherein said encapsulated references of at least one of said encapsulated data instances are unique and said encapsulated references of at least two of said encapsulated data instances are generally identical.

19. The data management system of claim 1 wherein said data instance centric architecture includes plurality of pre-existing encapsulated data instances, and said plurality of pre-existing encapsulated data instances have established associations, and at least one new encapsulated data instance is associated with at least one of said pre-existing encapsulated data instances.

20. The data management system of claim 1 wherein:

said data instance centric architecture includes a plurality of pre-existing encapsulated data instances; said encapsulated data instances have established associations; and wherein any of said pre-existing encapsulated data instances can be removed disassociated from other pre-existing associated encapsulated data instances.

21. The data management system of claim 1 wherein:

said data instance centric architecture includes a plurality pre-existing encapsulated data instances; said encapsulated data instances having established associations; wherein new associations between at least two pre-existing encapsulated data instances can be added.

22. The data management system of claim 1 wherein:

said data instance centric architecture includes a plurality of pre-existing encapsulated data instances; said encapsulated data instances having established associations; and wherein some of said pre-existing associations between said pre-existing encapsulated data instances can be removed.

23. The data management system of claim 1 further comprising:

a. finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing said known encapsulated data instances representing said selection criteria;

b. accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

c. using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

d. retrieving said specific unknown encapsulated data instances.

24. The method of claim 23 wherein:

a. said encapsulated references are embodied as logical indexes in a plurality of dimensions;

b. each of said dimensions corresponds to a type of association; and

c. said accessing further comprises accessing said encapsulated references from said dimensions specified in said selection criteria.

25. The method of claim 23 wherein:

said encapsulated references are 'm' dimensional logical indexes each of which uniquely identifies and encodes the location of said associated encapsulated data instances; and

further comprising filtering said encapsulated references by Boolean operations on at least one of said 'm' dimensional logical indexes.

26. The method of claim 24 wherein:

said encapsulated references are 'm' dimensional logical indexes each of which uniquely identifies and encodes the location of said associated encapsulated data instances; and

further comprising filtering said encapsulated references by Boolean operations on at least one of said 'm' dimensional logical indexes.

27. The method of claims 23 wherein said Boolean operations further comprise: basic mathematical operators which result in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

28. The method of claims 24 wherein said Boolean operations further comprise: a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

29. The method of claims 25 wherein said Boolean operations further comprise: a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

30. The method of claims 26 wherein said Boolean operations further comprise: a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

31. The system of claim 1 wherein said encapsulated data instances have attributes of a user interface.

32. The system of claim 31 wherein said attributes of a user interface are selected from a group of user views, display elements, and data access methods.

33. The system of claim 1 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

34. The system of claim 33 wherein said encapsulated references of different said encapsulated data instances are compared such for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

35. The system of claim 34 wherein said encapsulated references of different said encapsulated data instances are stored in an order based on value and are compared such for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

36. The system of claim 33 wherein:

a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance;

wherein each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances; and

wherein said logical index is 'm' dimensional, and has 'n' bits per dimension;

said encapsulated references of different said encapsulated data instances are used by comparing such for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

37. The system of claim 33 wherein:

each of said at least one dimensions has a plurality of said encapsulated references; and

said encapsulated references of different of said encapsulated data instances are stored in an order based on value and are compared for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

38. The system of claim 33 wherein comparisons are done in parallel using a hardware apparatus comprising comparator connected arrays of shift registers attached as a port to a memory bus in said computing environment.

39. The system of claim 38 further comprising using a hardware apparatus comprising logic circuits and a concatenated series of shift registers which determine search results.

40. The system of claim 1 further comprising:
encapsulated data instances representing ASCII characters;
said common fundamental data structures containing said encapsulated data instances representing ASCII characters also contain encapsulated references to encapsulated data instances containing said corresponding ASCII characters; and
said common fundamental data structures containing said encapsulated data instances containing said corresponding ASCII characters also contains encapsulated references to said encapsulated data instances representing corresponding ASCII characters.

41. The system of claim 40 wherein said encapsulated references with a given ASCII character data instance are references to other encapsulated data instances containing said ASCII characters based on position of said ASCII characters in the sequence of occurrence of said ASCII characters in said encapsulated data instances.

42. The system of claim 1 further comprising:
said encapsulated data instances representing Unicode characters;
said common fundamental data structures containing said encapsulated data instances representing Unicode characters also contain encapsulated references to encapsulated data instances containing said corresponding Unicode characters; and

said common fundamental data structures containing said encapsulated data instances representing Unicode characters also contains encapsulated references to said data instances representing corresponding Unicode characters.

43. The system of claim 42 wherein said encapsulated references with a given Unicode character data instance are references to other data instances containing said Unicode characters based on position of said Unicode characters in the sequence of occurrence of said Unicode characters in said encapsulated data instances.

44. The system of claim 1 wherein:

said encapsulated data instances comprise encapsulated data instances representing a token set of any data type;

said common fundamental data structures containing said data instances representing a token set of any data type also contain encapsulated references to encapsulated data instances containing said corresponding token set of any data type; and

said common fundamental data structures containing said encapsulated data instances representing token set of any data type also contains encapsulated references to said encapsulated data instances representing corresponding token set of any data type.

45. The system of claim 44 wherein said encapsulated references for a given token set of any data type data instance are references to other encapsulated data instances containing said token set of any data type based on position of token set of any data

type in the sequence of occurrence of said token set of any data type in said encapsulated data instances.

46. The system claim 45 wherein:

said token set is selected from a group of a set of graphic descriptors, a set of colors, a set of shapes, a set of glyphs, a set of waveforms, a set of frequency values, a set of audio frequency values, a defined set of symbols, and real numbers.

47. The data management system of claim 1 wherein:

a. the common fundamental data structure is application independent and is generally the same for all of said data instances;

b. finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

c. accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

d. using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

e. retrieving said specific unknown encapsulated data instances.

48. The system of claim 47 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

49. The data management system of claim 1 wherein:

at least one of said encapsulated references is a reference to a encapsulated data instance in another computing environment; and

a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to an encapsulated data instance.

50. The data management system of claim 1 wherein:

a. said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical;

b. finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

c. accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

d. using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

e. retrieving said specific unknown encapsulated data instances.

51. The data management system of claim 1 wherein:

said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical; and

searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

52. A data management system in a computing environment comprising:

a. a data instance centric architecture;

b. where each data instance is encapsulated in a common fundamental data structure;

c. where said common fundamental data structure also encapsulates references to associated separately encapsulated data instances;

d. a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance;

e. each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances;

f. said logical index is 'm' dimensional, and has 'n' bits per dimension; and

g. said encapsulated references are in at least one dimensions; and

each of said at least one dimensions corresponds to a type of association.

53. The data management system of claim 52 wherein the common fundamental data structure is application independent and is generally the same for all of said data instances.

54. The data management system of claim 53 further comprising:

a. finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

b. accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

c. using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

d. retrieving said specific unknown encapsulated data instances.

55. The system of claim 54 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

56. The data management system of claim 55 wherein at least one of said encapsulated references is a reference to a encapsulated data instance in another computing environment.

57. The data management system of claim 56 wherein said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instances are generally identical.

58. The system of claim 57 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

59. The data management system of claim 52 further comprising:

a. finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

b. accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

c. using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

d. retrieving said specific unknown encapsulated data instances.

60. The system of claim 52 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

61. The data management system of claim 52 wherein at least one of said encapsulated references is a reference to a encapsulated data instance in another computing environment.

62. The data management system of claim 52 wherein said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical.

63. A method to coordinating physical memory addressing and logical memory addressing in an encapsulated data instance centric architecture comprising:
corresponding to each encapsulated data instance there is a logical reference to said encapsulated data instance;
encapsulating said logical reference to said encapsulated data instance in a first container;
relating said logical reference in said first container with a physical reference to a location where said encapsulated data instance is stored in a physical storage medium;
encapsulating said physical reference in a second container; and
relating said physical reference in said second container with said logical reference to said encapsulated data instance in said first container.

64. The method of claim 63 wherein said container is a fundamental data structure.

65. The method of claim 63 wherein said physical reference is 'n' dimensional, the number of dimensions and the number of bits in each dimension correspond to the structure of said physical storage medium.

66. The method of claim 65 wherein using said physical reference to calculate an address in said physical storage medium.

67. The method of claim 63 further comprising:
coordinating a plurality of data instances; and
encapsulating a plurality of said logical references and a plurality of said physical references in respective said first and second containers.

68. The method of claim 63 further comprising sorting said plurality of said logical references in said first containers.

69. The method of claim 68 further comprising sorting said plurality of said physical references in said second containers.

70. A method for managing data storage in a data instance centric architecture having a plurality of variable length data instances comprising:
storing said data instances generally sequentially in a physical storage medium;
storing each data instance in a respective allocated space;
updating one of said data instances;

integrating said updated data instance into said physical storage medium by determining the amount of physical space that is needed to store said updated data instance;

when said physical space is equal to or less than said respective allocated space then storing said updated data instance in the said respective allocated space;

when said physical space is greater than said respective allocated space then identifying at least one physically adjacent data instance having an aggregate allocation space equal to or greater than the difference between said physical space and said respective allocated space; and

writing the said updated data instance to said physical storage medium at an updated location based on the size and number of said adjacent data instances.

71. The method of claim 70 for managing data storage further comprising:

writing the said updated data instance to said physical storage medium in an updated respective allocated space based on the size and number of said adjacent data instances.

72. The method of claim 70 for managing data storage further comprising:

moving said at least one physically adjacent data instance to a location after a last stored data instance; and

writing the said updated data instance to said physical storage medium to a location of said at least one physically adjacent data instance and said respective data instance.

73. The method of claim 70 for managing data storage further comprising:

moving said at least one physically adjacent data instance to a location after a last stored data instance; and

writing the said updated data instance to said physical storage medium in an updated allocation space equal to the aggregate of said respective allocated space and the allocated space of the said moved at least one physically adjacent data instances.

74. The method of claim 73 for managing data storage wherein the said updated allocation space is greater than the said physical space of the said updated data instance.

75. The method of claim 70 for managing data storage wherein said at least one physically adjacent data instance occur sequentially after said updated data instance.

76. The method of claim 70 for managing data storage wherein said at least one physically adjacent data instance occur sequentially before said updated data instance.

77. The method of claim 70 for managing data storage wherein said at least one physically adjacent data instance occur sequentially both before and after said updated data instance.

78. The method of claim 70 for managing data storage wherein:

whichever of said at least one physically adjacent data instance has an allocated space that is closest to and greater than the said difference between said physical space and said respective allocated space of the said updated data instance the said at least one physically adjacent data instance is moved to a location after a last stored data instance;

writing the said updated data instance to said physical storage medium to a location of said at least one physically adjacent data instance and said respective data instance; and

writing the said updated data instance to said physical storage medium in an updated allocation space equal to the aggregate of said respective allocated space and the allocated space of the said moved at least one physically adjacent data instances.

79. The method of claim 70 for managing data storage wherein when said physical space of said updated data instance is less than the allocated space of said at least one physically adjacent data instance then the updated data instance is moved to a location after a last stored data instance.

80. The method of claim 79 for managing data storage wherein the said respective allocated space of the said moved updated data instance is added to the allocated space of the physically adjacent data instance sequentially before said updated data instance.

81. The method of claim 70 for managing data storage wherein at least one of said allocated space is greater than the size of the respective data instance.

82. A method to convert a non-data instance centric database to a data instance centric database comprising:
creating data instances in said data instance centric database representing elements of said non-data-instance centric database schema and data elements of said non-data-instance centric database; and
create associations amongst the said data instances in said data centric database representing the relationships between said data elements and said schema elements of the non-data-instance centric database.

83. The method of claim 82 wherein said converting is through a software agent which is a data instance in said data instance centric database.

84. The method of claim 82 wherein said non-data instance centric database includes a flat file.

85. A data management system comprising:
one or more items;
wherein each of said items encapsulates a data instance; and
wherein items which are associated with each other encapsulate mutual references to each other.

86. The data management system of claim 85 wherein each of said items is represented in a fundamental data structure.

87. The data management system of claim 85 wherein each of said items has a unique reference associated therewith.

88. The data management system of claim 87 wherein said unique reference also serves as an index to physically locate said data instance associated with each of said items.

89. The data management system of claim 85 wherein said references to associated items are arranged in sets defining the type of association between said item and each of said other items referenced in said set.

90. The data management system of claim 87 wherein each of said references is an "m" dimensional index, each of said dimensions being "n" bits in length.

91. The data management system of claim 90 wherein "m" is 4 and "n" is 30.

92. The data management system of claim 85 wherein said items may act as containers for one or more other member items.

93. The data management system of claim 92 wherein membership of an item within a container item is indicated by an identity in one or more of said "m" dimensions in said logical index of said container item and each of said member items.

94. The data management system of claim 85 wherein each of said items may encapsulate embedded elements.

95. The data management system of claim 94 wherein said embedded elements are references to other items.

96. The data management system of claim 85 wherein said data instances may contain data of any type.